CS 121 Final Project Reflection

Due: March 16th, 11:59PM PST

This reflection document will include written portions of the Final Project. Submit this as **`reflection.pdf`** with the rest of the required files for your submission on CodePost.

For ease, we have highlighted any parts which require answers in **blue**.

Student name(s): **Travis Xiang, Rohan Iyer**

Student email(s): **txiang@caltech.edu , riiyer@caltech.edu**

Part L0. Introduction

Answer the following questions to introduce us to your database and application; you may pull anything relevant from your Project Proposal and/or README.

Database/Application Overview

What application did you design and implement? What was the motivation for your application? What was the dataset and rationale behind finding your dataset?

Database and Application Overview Answer (3-4 sentences) :

We implemented an application that will help select the "Team of the Year" for a given nationality. We did this as we wanted to be able to select the top performing players for each country at once. The current applications for TOTYs are solely based on overall performance, which is not specific enough for managers, for example. Our application allows for the selection of players in a more specific setting.

Data set (general or specific) Answer:

The dataset we chose was the FIFA 20 players dataset. We did this as this has information on all the professional players. Moreover, this inspired the TOTY, as FIFA was the first to do this.

***Client user(s) Answer*:**

The intended clients are those who want to access various attributes about the players in the video game and see how manipulations in the attributes of players affect their ability to make the TOTY.

Admin user(s) Answer:

The admin users are managers for clubs who want to see how their players rank and can change who is on their team as a projection for how their club will do next season.

Part A. ER Diagrams

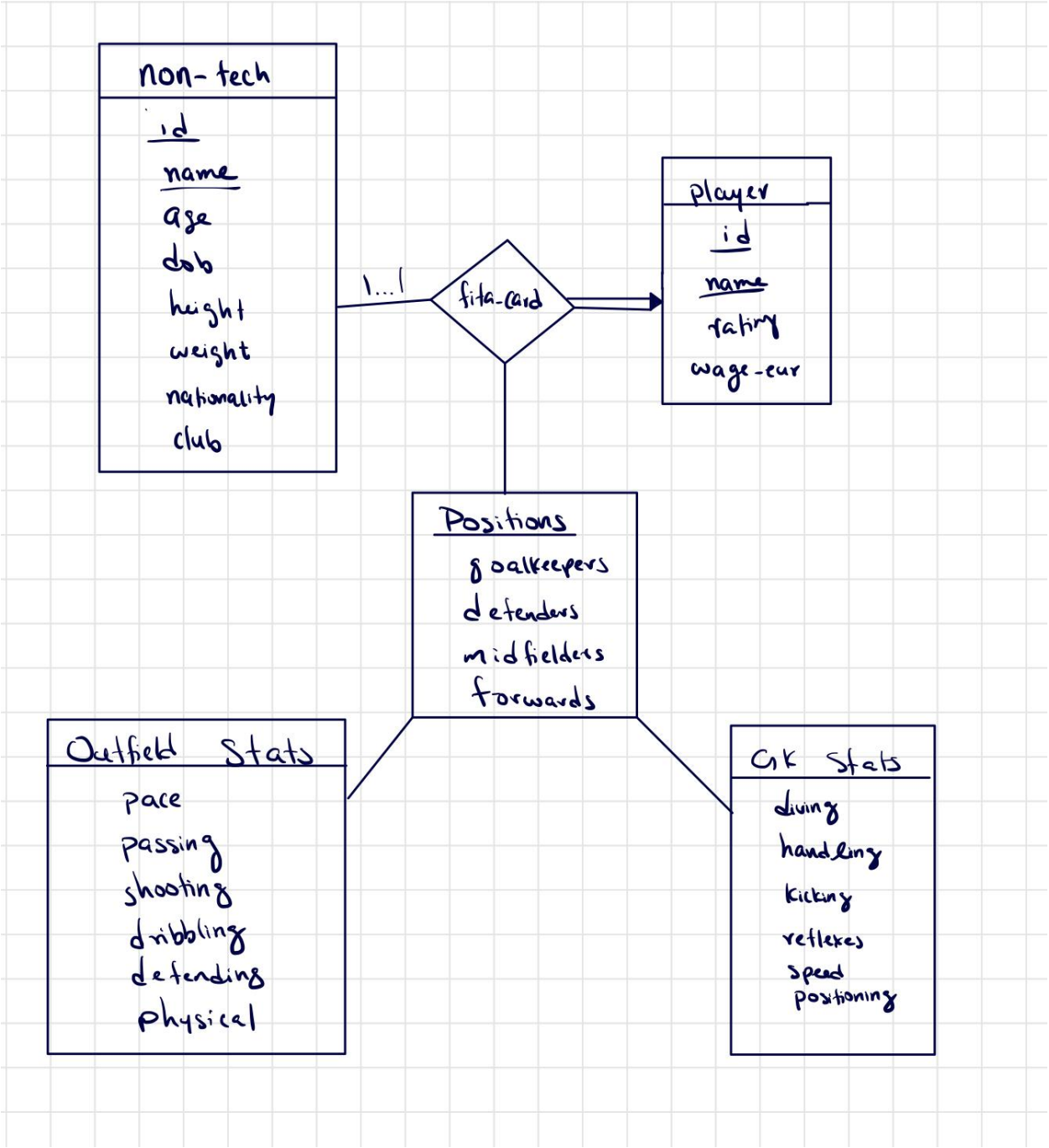
As we've practiced these past few weeks, the ER model is essential for designing your database application, and we expect you to iterate upon your design as you work through the ER and implementation steps. In this answer, you should provide a full ER diagram of your system. Your grade will be based on correct representation of the ER model as well as readability, consistency, and organization.

Notes: For this section **only**, we will allow (and encourage) students to share their diagrams on Discord (**#er-diagram-feedback**) to get feedback from other students on their ER diagrams given a brief summary of your dataset and domain requirements. This is offered as an opportunity to test your ER diagrams for accuracy and robustness, as another pair of eyes can sometimes catch constraints that are not satisfied or which are inconsistent with your specified domain requirements.

Requirements:

- Entity sets, relationship sets, and weak entity sets should be properly represented (also, do not use ER symbols not taught in class)
- Mapping cardinalities should be appropriate for your database schema, and in sync with your DDL
- Participation constraints should be appropriate and in sync with your DDL (total, partial, numeric)
- Use specialization where appropriate (e.g. *purchasers* and *travelers* inheriting from a *customers* specialization in A6)
- Do not use degrees greater than 3 in your relationships, do not use more than one arrow in ternary relationships.
- Use descriptive attributes appropriately
- Underline primary keys and dotted-underline discriminators
- Expectations from A6 still apply here
- Note: You do not need ER diagrams for views

ER Diagrams:



Part B. DDL (Indexes)

As mentioned in Part B, you will need to add at least one index at the bottom of your **`setup.sql`** and show that it makes a performance benefit for some query(s).

Here, describe your process for choosing your index(es) and show that it is used by at least one query, which speeds up the performance of the same query on a version of the same table without that index. You may find lec14-analysis.sql and Lecture 14 slides on indexes useful for strategies to choose and test your indexes. **Remember that indexes are already created in MySQL for PKs and FKs, so you should not be recreating these.**

Index(es):

We created an index on player(rating, wage) as well as on nontechnical_attributes (age, nationality).

Justification and Performance Testing Results:

We chose these as these were the attributes (besides id and name) that were accessed most frequently. This can be seen by our application, which sorts based on nationality, and our UDFs, which heavily rely on wage, age, and rating.

Part C. Functional Dependencies and Normal Forms

Requirements (from Final Project specification):

- Identify *at least 2 non-trivial functional dependencies* in your database
- Choose and justify your decision for the normal form(s) used in your database for at least 4 tables (if you have more, we will not require extra work, but will be more lenient with small errors). BCNF and 3NF will be the more common NF's expected, 4NF is also fine (but not 1NF).
    - Your justification will be strengthened with a discussion of your dataset breakdown, which we expect you to run into trade-offs of redundancy and performance.
- For two of your relations having at least 3 attributes (each) and at least one functional dependency, prove that they are in your chosen NF, using similar methods from A7.
    - If you have identified functional dependencies which are not preserved under a BCNF decomposition, this is fine
- Expectations from A7 still apply here.

Functional Dependencies:

1. Given the relation: player(id, name, rating, wage)
    a. There is a functional dependency of: (rating, wage) → (id, name)
2. Given the relation: nontechnical_attributes(id, name, age, dob, height, weight, nationality, club)
    a. There is a functional dependency of: (id, name, dob, nationality) → club

Normal Forms Used (with Justifications):

We decided to use the BCNF since the database is not particularly large with only around 11,000 entries. Therefore, we rationalized that the accuracy and correctness was more important than runtime since we would get marginal returns of speed improvements on a small database and potentially sacrifice correctness.

NF Proof 1:

R1(id; name, dob, nationality) R2(id; age, height, weight, club)

- ➔ No Functional Dependencies in R2
- ➔ Primary Key: id which is also a superkey so in BCNF
    - ◆ R1 = (id; name, dob, nationality)
    - ◆ R2 = (id; age, height, weight, club)
    - ◆

NF Proof 2:

R1 (rating, wage; id, name) R2(rating, wage)

- ➔ No Functional Dependencies in R2

➔ Primary Key: (rating, wage) which is also a superkey so in BCNF
➔ Dependencies Lost: id → (name, rating, wage)

Part G. Relational Algebra

Requirements (from Final Project specification, Part G):

- Minimum of 3 non-trivial queries (e.g. no queries simply in the form **`SELECT <x> FROM <y>`**)
- At least 1 group by with aggregation
- At least 3 joins (across a minimum of 2 queries)
- At least 1 update, insert, and/or delete
    - This may be equivalent to said SQL statements elsewhere (e.g. queries or procedural code), but are not required to be; in other words, you can write these independent of other sections
- Appropriate projection/extended projection use
- Computed attributes should be renamed appropriately
- Part of your grade will come from overall demonstration of relational algebra in the context of your schemas; obviously minimal effort will be ineligible for full credit; it is difficult to formally define "obviously minimal", but refer to A1 and the midterm for examples of what we're looking for
- Above each query, briefly describe what it is computing; we will use this to grade for correctness based on what the query is supposed to compute; lack of descriptions will result in deductions, since we have no idea otherwise of what the query is intended to do.

Below, provide each of your RA queries following their respective description.

RA #1, #3 correspond to queries #1, #3

Relational Algebra

1) Select the top goalkeeper's name, club, nationality, avg-rating based on the average of each individual goalkeeping stat.

$$\text{rating\_table} \leftarrow \Pi_{id,\ \frac{1}{6}(\text{diving}+\text{handling}+\text{kicking}+\text{reflexes}+\text{speed}+\text{positioning})\ \text{as avg-rating}}(\text{goalkeepers})$$

$$\text{player\_desc} \leftarrow \Pi_{\text{name, club, nationality, avg-rating}}\left(\left(\text{player} \bowtie \text{nontechnical\_attributes} \bowtie G_{\max(\text{avg-rating})}(\text{rating-table})\right)\right)$$

2) Choose the best midfielders with the highest passing stat who are at least twice as fast as the slowest midfielder.

$$\text{slowest\_mid} \leftarrow G_{\text{sum(pace)}}\left(\Pi_{\text{pace}}\left(\text{midfielders} \bowtie G_{\min(\text{pace})}(\text{midfielders})\right)\right)$$

$$\text{fast\_enough} \leftarrow \Pi_{\text{name, pace, passing}}\left(\sigma_{\text{midfielders.pace} > 2 \cdot \text{slowest\_mid}}(\text{midfielders} \bowtie \text{player})\right)$$

$$\text{best\_mid} \leftarrow \Pi_{\text{name, pace, passing}}\left(\text{fast\_enough} \bowtie G_{\max(\text{passing})}(\text{midfielders})\right)$$

3) Insert the fastest forward into the goalkeepers table

$$\text{pace\_merchant} \leftarrow \Pi_{id}\left(\text{forwards} \bowtie G_{\max(\text{pace})}(\text{forward})\right)$$

$$\text{goalkeepers} \leftarrow \text{goalkeepers} \cup \{\text{pace\_merchant} \times \{(1,1,1,1,1,1)\}\}$$

Part L1. Written Reflection Responses

Challenges and Limitations

List any problems (at least one) that came up in the design and implementation of your database/application (minimum 2-3 sentences)

Answer:

One problem was our initial proposal was too ambitious. In practice, we had to limit what we could do to reasonably finish within the deadline. Some features, such as selecting the most promising individuals, would require functions with more involved computations to predict the potential of each player.

Future work

If you are particularly eager for a certain application (have your own start-up in mind?), it is easy to over-scope a final project, especially one that isn't a term-long project. You can list any stretch goals you might have "if you had the time" which staff can help give feedback on prioritizing (2-3 sentences).

Answer:

In the future, it would be possible to predict the top performers on a more frequent basis, such as a weekly basis given current performance. We would also be able to implement the most promising players, as described above and in the proposal, which would have been a cool feature. However, we figured that the features we implemented were more important than these, so we focused on those first.

Collaboration (Required for Partners)

This section is required for projects which involve partner work. Each partner should include 2-3 sentences identifying the amount of time they spent working on the project, as well as their specific responsibilities and overall experience working with a partner in this project.

Partner 1 Name: **Travis Xiang**

Partner 1 Responsibilities and Reflection:

Travis did parts B, D, E, F, H, and helped work on I, J, K, and L. The amount of time spent was the same as Rohan's, which was around 20 hours. The overall experience was good and the project was fun, albeit a bit time consuming.

Partner 2 Name: **Rohan Iyer**

Partner 2 Responsibilities and Reflection:

Rohan did parts A, C, and G, and helped work on I, J, K, and L. The amount of time spent was the same as Travis', which was around 20 hours. Personally, I found that the coding was a bit easier than the theoretical parts and also the time limits were a bit off because of the time needed to set up the data and debug as well.

Other comments

This is the first time CS 121 has had a Final Project, and we would appreciate your feedback on whether you would recommend this in future terms, as well as what you found most helpful, and what you might find helpful to change.

Answer:

We thought the project was better than having an exam, as the collaboration aspect of having a partner was enjoyable. We also enjoyed creating our own project. We would find it helpful to have the time estimates be more accurate, but we recognize that this is a difficult task.